# A Multi-Agent based Decision Mechanism for Incident Reaction in Telecommunication Network

Bonhomme, Cédric; Feltus, Christophe; Khadraoui, Djamel

*DOI:*
10.1109/AICCSA.2010.5587036

*Publication date:*
2010

*Document Version*
Early version, also known as pre-print

# A Multi-Agent based Decision Mechanism for Incident Reaction in Telecommunication Network

Cédric Bonhomme, Christophe Feltus and Djamel Khadraoui
Centre for IT Innovation
Public Research Centre Henri Tudor
29, Avenue John F. Kennedy, L-1855 Luxembourg
Email: cedric.bonhomme@tudor.lu

***Abstract*—The main objective of this paper is to provide a global architectural and decision support solution built on the requirements for a reaction after alert detection mechanisms in the frame of information systems security and more particularly applied to telecom infrastructures security. These infrastructures are distributed in nature, therefore the targeted architecture is developed in a distributed perspective and is composed of three basic layers: low level, intermediate level and high level. The low level constitutes the interface between the main architecture and the targeted infrastructure. The intermediate level is responsible of correlating the alerts coming from different domains of the infrastructure and to smartly deploy the reaction actions. The architecture is elaborated using the multi-agents system that provides the advantages of autonomous and interaction facilities, and has been associated to the OntoBayes model for decision support mechanism. This model helps agents to make decisions according to preference values and is built upon ontology based knowledge sharing, Bayesian networks based uncertainty management and influence diagram based decision support. The major novelty of this Decision Support System is the layered view of the infrastructure thanks to MAS architecture, which enables the decision making with different levels of knowledge. The proposed approach has been successfully experimented for data access control mechanism.**

***Index Terms*—security; decision system; reaction; distributed network; bayesian network; multi-agents system.**

## I. Introduction

Today information systems and mobile computing networks are more widely spread and mainly heterogeneous. This basically involves more complexity through their opening, their interconnection and their ability to make decisions [9]. Consequently, this has a dramatic drawback regarding threats that could occur on such networks via dangerous attacks (i.e.: introduction of a malicious code or evil-minded modication of a the DNS configuration file) [26]. This continuously growing amount of carry out malicious acts encompasses new and always more sophisticated attack techniques, which are actually exposing operators as well as the end user.

State of the art in terms of security reaction is limited to products that detect attacks and correlate them with a vulnerability database but none of these products are built to ensure a proper reaction to attacks in order to avoid their propagation and/or to help an administrator to deploy the appropriate reactions [19], [23]. In the same way, [22] says that at the individual host-level, intrusion response often includes security policy reconfiguration to reduce the risk of further penetrations but doesn't propose another solution in term of automatic response and reaction. It is the case of CISCO based IDS material providing mechanisms to select and implement reaction decision.

Information security management and communication systems is actually in front of many challenges [12] due to the fact that it is very often difficult to establish central or local permanent decision capabilities, have the necessary level of information, quickly collect the information, which is critical in case of an attack on a critical system node, or launch automated counter measures to quickly block a detected attack.

Based on that statements, it appears crucial to elaborate a strategy of reaction after detection against these attacks. Our previous work around that topic has provided first issues regarding that finding and has been somewhat presented in [12] and [15]. These papers have proposed an architecture to highlight the concepts aiming at fulfilling the mission of optimizing security and protection of communication and information systems which purpose was to achieve the following:

- Reacting quickly and efficiently to any simple attack but also to any complex and distributed ones;
- Ensuring homogeneous and smart communication system configuration, that are commonly considered and the main sources of vulnerabilities.

One of the main aspects in the reaction strategy consists of automating and adapting policies when an attack occurs. In scientific literature a large number of definitions for policy and conceptual model exist. The most famous are Ponder [10], Policy Description Language [5] and Security Policy Language [2]. For the purpose of that paper, we prefer the one provided in [10]: Policies are rules that govern the behavior of a system.

The provided policy adaptation is considered as a regulation process. The main steps of the policy regulation are described in Figure 1, which shows the process that takes the business rules as input, and maps them onto technical policies. These technical policies are deployed and instantiated on the infrastructure in order to have a new state of temporary network security stability adapted to the ongoing attack. This policy regulation is thereafter achieved in modifying/adding new policy rules to reach a new standing (at least up to the next network disruption) policy based on the observation of the systems current situation.

In this paper, we focus our work on policy deployment and on policy modification decision-reaction challenges as highlighted in the rounded rectangle of Figure 1. This twofold challenge has already been addressed by other researches like in [28]. Torrellas explains that facilitating timely decision-making may achieve much greater productivity benefits by engineering network security systems using multi-agents. In [30], Yu developed the concepts of tele-service and proposed an implementation of an e-maintenance platform based on a Multi-Agent System (MAS). Yu explained how a Case-Based Reasoning [1] method may be used to improve the autonomous decision-making ability. Others works propose rather similar solutions like [21], [8] but none are explicitly dedicated to the management of security alerts reaction in the field of open networks.

Consequently, the paper propose a system that combine a reaction mechanism with the decision support. Such a problem has never been addressed before the recently emerging agent-based applications for reaction after detection infrastructures as presented in [16].

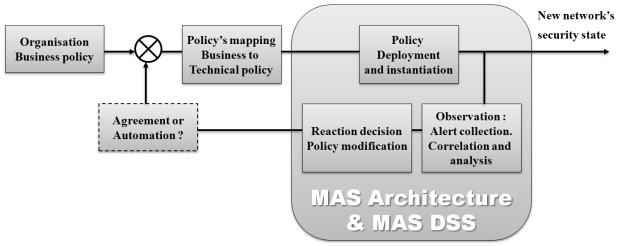


Fig. 1: Policy regulation

The next section introduces the MAS architecture. Section 3 exposes the decision support system as well as its combination with the MAS, and the last section concludes the paper.

## II. Multi-Agent System architecture

MAS is composed of several agents, capable of mutual interaction. The interaction can be in the form of message passing or producing changes in their common environment. Agents are pro-actively, reactively and socially autonomous entities able to exhibit organized activity, in order to meet their design objectives, by eventually interacting with users. An agent is collaborative by being able to commit itself to society and/or another agent.

An agent encapsulates a state and a behavior and provides moreover a number of facilities such as: control of its behavior, the ability to decide even if external events influence its decision, the possibility to exert its control in various manners (reactively, directed by goals, socially). Moreover, MAS have several control flows while a system with objects has a priori only one control flow.

The agents also have global behavior within the MAS, such as the cooperation (agents share the same goal), collaboration (agents share intermittently the same goal) or competition (incompatible goals between agents).

To manage several different systems, due to their location, their business domain or their organization type, a distributed system is appropriate. Furthermore, a distributed solution brings some autonomy to the managed systems. Robustness, survivability and availability are also impacted.

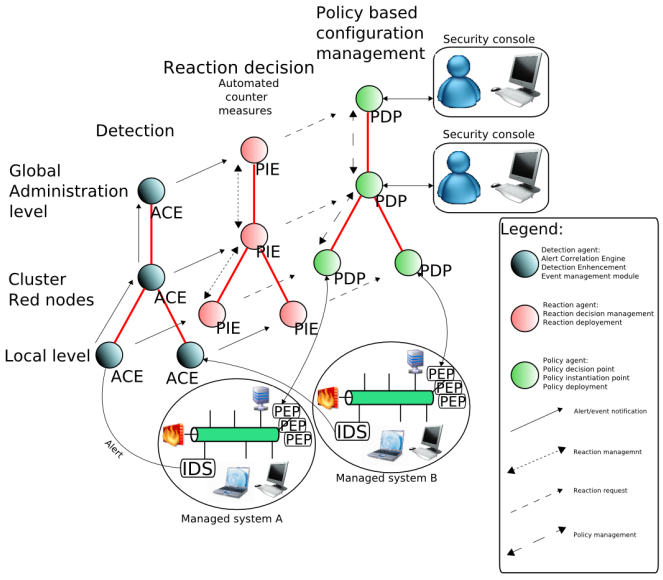


Fig. 2: Reaction architecture overview

The distributed architecture introduced in this paper is composed by several components, called “operators”, which have different responsibilities. Those operators are organized in two dimensions, as presented in Figure 2.

The vertical dimension, structured in layers relative to the managed network organization, allows adding abstraction in going upward. Indeed, the lowest layer is closed to the managed system and thus being the interface between the targeted network and the management system. The higher layer exposes a global view of the whole system and is able to take some decisions based on a more complete knowledge of the system, business, and organization. Intermediate levels (1 to n-1) guarantee flexibility and scalability to the architecture in order to consider management constraints of the targeted infrastructure. Those middleware levels are optional but allow the system to be better adapted to the complexity of a given organization and the size of the information system.

The horizontal dimension, containing three basic components, is presented in Figure 2 and its three main phases are described below:

1) *Alert*: Collect, normalize, correlate, analyze the alerts coming from the managed networks and represent an intrusion or an attack. If the alert is confirmed and coherent, it is forwarded to the reaction decision component. (Alert Correlation Engine-ACE).
2) *Reaction Decision*: Receive confirmed alerts for which a reaction is expected. Considering the knowledge of: policy, the systems' organization and specified behavior,

these components decide if a reaction is needed or not and define he reaction, if there is any. The reaction will be modification(s), addition(s) or removal(s) of current policy rules. (Police Instantiation Engine-PIE).

3) *Reaction*: Instantiation and deployment of the new policies, on the targeted networks. The deployment (Policy Deployment Point PDP) and enforcement (Policy Enforcement Point PEP) of these new policies, lead to a new security state of the network.

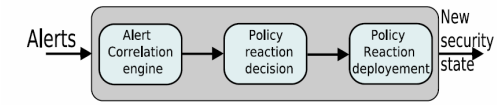


Fig. 3: The three basic components

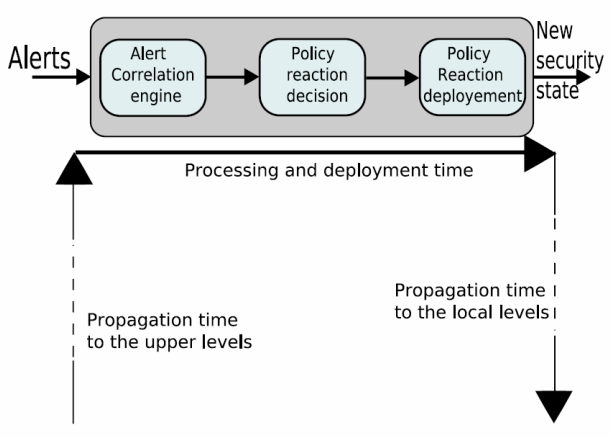


Fig. 4: Response time

An issue is raised considering which layer is allowed to take a decision reaction: only one layer, two, several, or all? If more than one layer can trigger a reaction on the same object(s), there will be a conflict issue. Thus, the system should be able to provide mechanisms to solve conflicts between several selected reactions. Another issue concerns the agreement: at which level should it be asked? A solution could be to ask at the same level (or at an upper one) that the reaction decision is made; this should be specified by the user. A possible solution is a distributed, vertically layered and hierarchical architecture. The layer's number could be adapted according to the managed systems organization. In our case, three layers are sufficient (local, intermediate and global). The reaction system is composed of three main parts: the alert management part, the reaction part and the police definition-deployment part. Three trees (alert, reaction and policy) could be placed side by side, as presented in Figure 2. These trees are alike but their operators have different functions. The alert tree collects the alerts with the local operators and correlate them in several steps, one step by layer. A certain response time is used by the system from intrusion detection to reaction application. This time is increased if the reaction process is propagated to upper layers, as presented in Figure 4. The global goal is of course to shorten it.

The next step of our research development is firstly the definition of a reaction engine that encompasses both, architecture components and the communication engine between these components. This engine is based on a message format and on a message exchange protocol based on standards such as [11]. Secondly, real cases are studied in order to experiment with the architecture and its associated protocol.

The message format is defined in XML format and is structured around a number of attributes that specify the message source, the message destination and the message type (alert, reaction, policy request, policy modification, policy modification validation, decision and synchronization). The protocol defines the exchange format and the workflow of messages between the architecture components. It encompasses a set a rules governing the syntax, semantics, and synchronization of communication. The technical requirements request the operator structure must be flexible in order to be able to reorganize itself, if an operator fails or disappears. Each operator also has to be autonomous in order to permit reorganization. Given these requirements, the use of a MAS appears as a solution to provide autonomy, flexibility and decision mechanisms to each operator that are consequently represented by agents.

As studied in the state of the art presented in [14], a set of agents could be managed and controlled through an organization. An organization is a set of agents playing roles, gathered in a normative structure and expecting to achieve some global and local objectives. Several models like the roles model, the tasks model, the interaction model or the norms models specify an organization.

In our context we need an interaction definition in order to specify communication protocols between agents representing operators. We also need roles in order to specify which agent will have to communicate or act in order to detect intrusions and then react. Based on this needs, the use of an electronic institution based on agents is one of the possibilities that we will investigate.

The main goal of the reaction policy enforcement engine is to apply policies in terms of specific concrete rules on "technical" devices (firewall, fileserver, and other systems named PEP). For that, we need means to make ACE, PIE, PDP and PEP interact and collaborate.

The multi-agents systems concept already defines architectures and models for autonomous agents organization and interaction. Existing platforms like JADE (Java Agent DEvelopment framework) [4], [3] implement agents' concepts as well as their ability to communicate by exchanging messages and the reaction components integration could be simplified. This is a solution, which will be detailed hereafter. The Foundation for Intelligent Physical Agents (FIPA) [13] promotes the success of emerging agent-based applications, services and equipment. It makes available internationally agreed specifications that maximize interoperability across agent based applications, services and equipment pursue this goal. This is realized through open international collaboration of member

organizations, which are companies and universities active in the agent field. FIPA's specifications are publicly available. They are not technologies for specific application, but generic technologies for different application areas, and not just independent technologies but a set of basic technologies that can be integrated by developers to make complex systems with a high degree of interoperability.

The used multi-agent framework is JADE. We base ourselves on a survey made in [6] to argue that this agent platform responds to the expectations in terms of agents' functionalities, security, performance, standardization, and secure communication between agents.

Figure 5 introduces the developed architecture. The flow is supposed to begin with an alert detected by the IDS, positioned on a network component. This alert is send to the ACE agent (or LAN ACE if it concerns a precise Local Area Network): This ACE agent confirms or not the alert to the PIE. This decision to confirm the alert is explained in section 3. Afterwards, the PIE decides to apply new policies or to forward the alert to an ACE from a higher layer (upper ACE). Its PIE agent sends the policies to the PDP agent, which decides which PEP is able to implement it in terms of rules or script on devices (firewall, fileserver, etc.) Then, the PDP agent sends the new policy to the concerned PEP agent that knows how to transform a policy into a rule or script understandable by the associated device (for example fileserver).

On Figure 5, dash dot lines stand for flow of messages encompassing alert or alert confirmation. Full lines stand for flow of messages containing policies information, and dot lines are reserved for decision support mechanisms.

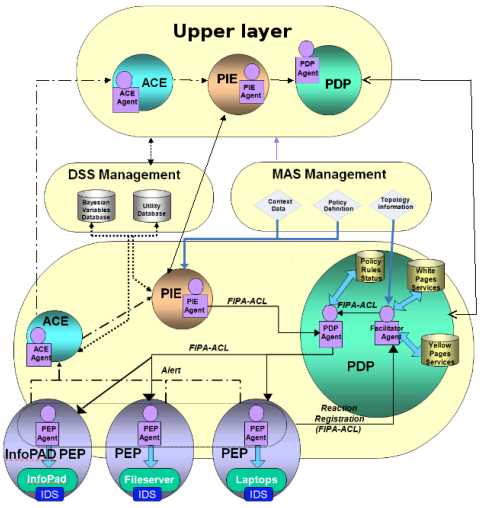


Fig. 5: Multi-Agent System reaction architecture

A focused analysis of the PDP shows that it is composed by several modules. For the multi-agent system point of view, the Component Configuration Mapper results from the interaction between the PDP agent and the Facilitator Agent while the Policy Analysis module is realized by the PDP agent. The Facilitator manages the network topology by retrieving PEP agents according to their localization (devices registered with IP address or MAC address) or according to actions they could apply and their type (firewall, file server, etc.). For that the Facilitator uses white pages and yellow pages services. The JADE platform already provides implemented facilitator and searching services. Besides, the use of a multi-agent system as the framework provides flexibility, and openness. Actually, when we decide to add a new PEP, we just have to provide its PEP Agent with the ability to concretely apply the policies that will register itself through the Facilitator, which will update the databases.

## III. Decision Support Architecture

Section 2 explains the developed MAS architecture that guarantees a telecommunication security incident reaction. This section explains the implementation of the decision mechanism for incident reaction, the main objective of this paper. For that reason the MAS architecture has voluntarily been explained before the Decision Support System (DSS) part because components of this architecture are used for the illustration of the DSS.

One important challenge of the DSS is the management of uncertainty. In our context uncertainty is defined as situation "*caused by a lack of knowledge about the environment when an agent need to decide the truth of statement*."

Decision is a process [20] and consequently, it may be represented using its input and its output. For the security incident reaction, inputs of the decision mechanism are for instance: the severity, duration and frequency of the alerts, the impact on the system, or the network criticality for the business whereas outputs are for instance: the escalation of the alert to upper ACE or its confirmation to the PIE.

As explained by Yang [29], the decision-making mechanism is composed of four pillars: Ontology, Bayesian Networks (BN), Influence Diagram (ID) and Virtual Knowledge Community (VKC). In the framework of that paper, the VKC will not be treated because the use of the 3 first pillars is enough to understand the decision mechanism. The approach preferred to design the decision mechanism is adapted from the research performed by Yangs thesis for the incident reaction through a MAS architecture. As a consequence our solution differs from and completes the Yang research since our DSS is illustrated by a real architecture for incident reaction that is really deployed in our research labs.

### A. Ontology

Ontology is the first pillar and is defined by a formal, explicit specification of a shared conceptualization [24]. Ontology may be categorized as domain ontology when it concerns concepts and their relations from a same and well-defined domain or top-level ontology when it concerns very general domain-independent concepts. Ontology is the most important pillar in that, it will be adapted to support the second pillar concerning the Bayesian Network and the third pillar concerning the Influence Diagram.

For the incident reaction system, ontology is defined using the Web Ontology Language (OWL). Resource Development

Frameworks (RDF) syntax is the most commonly used method to model information or meta-concepts in OWL. It may be implemented in web resources and is structured based on the triple (object, subject, predicate). Figure 6 illustrates RDF graph. Both, object and subject are resources whereas predicate is an attribute or a relation used to describe a resource.

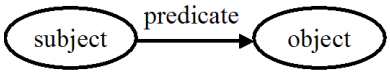


Fig. 6: RDF graph

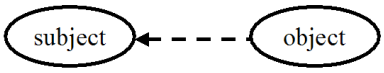


Fig. 7: Dependency graph

In parallel to the MAS architecture developed in section 3, we need a DSS to decide the transfer of an alert from the IDS to the in-LAN ACE[1], for the forward of that alert to an upper ACE, and for the confirmation of the alert to the PIE. This is formalized using OWL as explained in Figure 8. On that figure, ovals stand for OWL class, solid arrow lines stand for RDF predicate, dash arrows for influence relations and rounded rectangles for set of domain value.

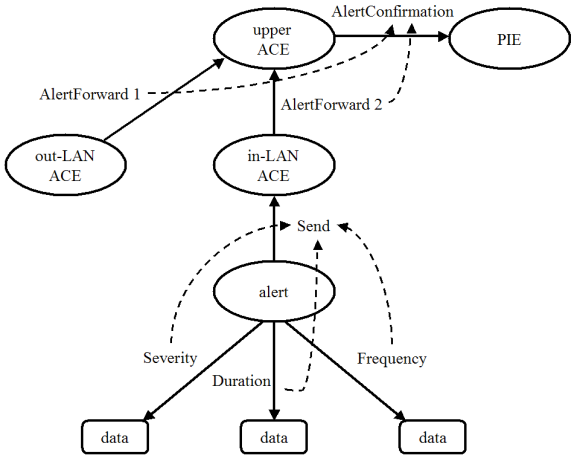


Fig. 8: Decision system for alert transfer using OWL

## B. OntoBayes

Ontology developed in the previous section permits to formalize the concept encompassed in the MAS architecture as well as their relations. However, at that the ontological level of formalization, uncertainty challenge remains unaddressed and decision mechanism remained needed for the agents to take the decision.

[1]ACE agent located in the Local Area Network where alert is sent.

OntoBayes is an extension of OWL with two features: Bayesian Networks (BNs) and Influence Diagram. BN address the uncertainty and ID support the decision mechanism process.

*1) Bayesian networks extension:* In probabilistic, Bayes Theorem is a simple mathematical formula used for calculating conditional probabilities [27]. It means that the calculations of probability depend on prior knowledge that could be considered as uncertain. I.e.: the probability of having a high impact on the system if we have an alert of medium severity. This probability is written P(alert.severity|system.impact).

The BNs extension of OWL introduces the parameters of that formula by specifies the following two perspectives: a qualitative perspective and a quantitative perspective. The qualitative perspective specifies the random variables explicitly as well as their dependencies and the later associates' quantitative information to those variables.

The specification of random variable and their dependency is performed by introducing the new OWL property element `<owl:ObjectProperty rdf.ID="dependsOn" />` and could be graphical represented as illustrated on Figure 7.

Accordingly, the qualitative extension may be represented by 2 Bayesian graph models (Figure 9) extracted from the OWL graph model from Figure 7.

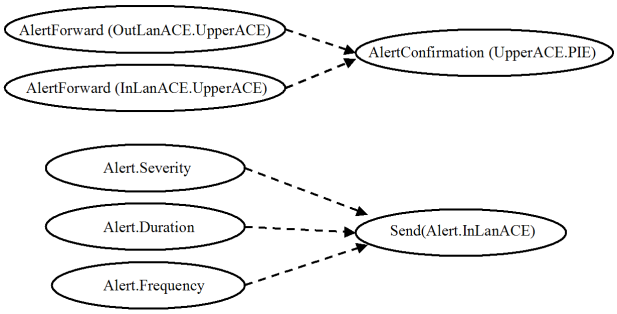


Fig. 9: Bayesian graph models

The ovals represent Bayesian variables and the arrows specify their relations. The graph is to be read i.e. 1.: The alert that is forwarded from the out-LAN ACE[2] to the network upper ACE has influence on the confirmation of the alert that is send from the upper ACE to the PIE. I.e. 2.: The severity of the alert has influence on the action to send an alert to the in-LAN ACE. The last examples may be translated using the new OWL `dependsOn` element as in Figure 10.

The quantitative extension is performed with the association of probability table to the Bayesian variables. In the case of the above example, the Table I provides the quantitative probability P(alert.severity|system.impact) and is represented on Figure 10 by the Bayesian variables database.

For example according to Bayes' formula for conditional probability the first line means that if the severity is low, the

[2]In opposition with in-LAN ACEs, out-LAN ACEs are agents located in others Local Area Networks than the LAN where alert is sent.

```
<owl:Class rdf:ID="alert.severity">
  <owl:Restriction>
    <owl:onProperty>
      <owl:ObjectProperty red:ID="dependsOn" />
    </owl:onProperty>
    <owl:hasValue rdf:resource="system.impact" />
  </owl:Restriction>
</owl:class>
```

Fig. 10: Dependency encoding

TABLE I: Bayesian variables value probability

| ProbCell | HasPParameters | HasPValue |
|---|---|---|
| Cell_1 | alert.severity=low\|system.impact=low | 0.6 |
| Cell_2 | alert.severity=medium\|system.impact=low | 0.3 |
| Cell_3 | alert.severity=high\|system.impact=low | 0.1 |
| Cell_4 | alert.severity=low\|system.impact=medium | 0.2 |
| Cell_5 | alert.severity=medium\|system.impact=medium | 0.5 |
| Cell_6 | alert.severity=high\|system.impact=medium | 0.3 |
| Cell_7 | alert.severity=low\|system.impact=high | 0.1 |
| Cell_8 | alert.severity=medium\|system.impact=high | 0.3 |
| Cell_9 | alert.severity=high\|system.impact=high | 0.6 |

probability that the impact is low will be relatively high. Of course if the severity is high the chance to have a low impact on the system is minimal (generally):

$$P[alert.severity = high \,|\, system.impact = low] = 0.1$$

In order to follow the law of total probability we must have :

$$\begin{aligned} &P[alert.severity = low \cap system.impact = low] \\ &+ P[alert.severity = medium \cap system.impact = low] \\ &+ P[alert.severity = high \cap system.impact = low] \\ &= 0.6 + 0.3 + 0.1 \\ &= 1 \end{aligned}$$

The conditional probability from Table I is encoded as follows (Figure 11):

```
<owl:Class rdf:ID="Alert">
  <CondProbDist rdf:ID="table_1">
    <hasPCell>
      <ProbC rdf:ID="Cell_1">
        <HasPValue rdf:Iddatatype="#float">0.6</HasPValue>
        <HasParameters rdf:datatype="#string">
          alert.severity=low|system.impact=low
        </HasParameters>
      </ProbC>
    </HasPCell>
    ...
  </CondProbDist>
</owl:Class>
```

Fig. 11: Bayesian variables value probability encoding

*2) Influence diagrams extension:* IDs extension aims at representing and analyzing a decisional model to support the decision-making process. The review of the literature that treats ID [17], [18] shows that decision mechanisms are composed by three types of nodes: 1) Chance nodes that represent variables that are not controled by the decision maker, 2) Decision nodes that represent choices available for the decision maker, and 3) Utility nodes that represent agent utility functions. Additionally, [25] explains that three type of arcs express the relationship between nodes: I) Information arcs (`isKnownBy`) that point out the information that is necessary for the decision maker, II) Conditional arcs (`influenceOn`) that point out the probabilistic dependency on the associated variable, and III) Functional arcs (`attributeOf`) that point out variables used by utility nodes as decision criteria.

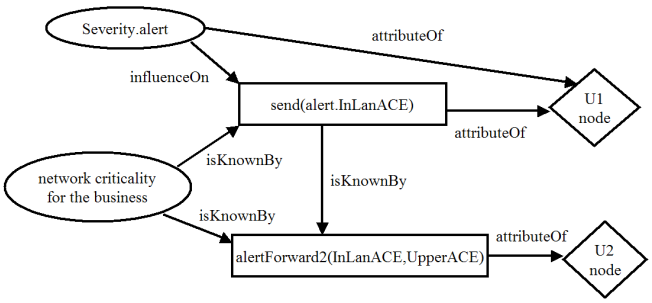


Fig. 12: IDs graph model of alert transfer

Based on that structure of decisional model, the alert transfer may be represented in Figure 12. Ovals stand for Chance nodes, rectangles stand for Decision nodes, and diamonds stand for Utility nodes. The information arc relates to all information observed to make a decision and the conditional arc relates to data issued from Chance node and considered as evidence for the Decision nodes.

Additionally, to make a decision, the agent that takes a decision needs to have its preferences quantified according to a set of attributes. In [7], Butler introduces the theory of multi-attribute utility (MAUT) to quantify a preference with numerical value. The best preference has the higher value whereas the worst has the lower one. To achieve that, the Utility node is associated with a utility table that gathers the preferences of all decision choices. Table II shows these preferences for the in-LAN ACE alert sending decision taking mechanism and is represented by the utility database in Figure 5.

TABLE II: Utility table for in-LAN ACE alert sending

| UtilityCell | HasUParameters | HasUValue |
|---|---|---|
| Cell_1 | send(alert.InLanACE)=yes\|severity.alert=low | -80 |
| Cell_2 | send(alert.InLanACE)=yes\|severity.alert=medium | 50 |
| Cell_3 | send(alert.InLanACE)=yes\|severity.alert=high | 100 |
| Cell_4 | send(alert.InLanACE)=no\|severity.alert=low | 80 |
| Cell_5 | send(alert.InLanACE)=no\|severity.alert=medium | 40 |
| Cell_6 | send(alert.InLanACE)=no\|severity.alert=high | -100 |

The Figure 13 shows the encoding of Table II utility table

for in-Lan ACE alert sending.

```
<owl:Class rdf:ID="send(alert.InLanACE)">
  <owl:Restriction>
    <owl:onProperty>
      <owl:ObjectProperty rdf:ID="attributeOf" />
    </owl:onProperty>
    <owl:hasValue rdf:resource=#U />
  </owl:Restriction>
  ...
  <rdfs:subClassOf>
    <owl:hasValue rdf:ID="DecisionNode" />
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="severity.alert">
    ...
  <rdfs:subClassOf>
    <owl:hasValue rdf:ID="ChanceNode" />
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="U1">
  <UtilityTable rdf:ID="table_1">
    <hasUCell>
      <UtilityCell rdf:ID="cell_1">
        <hasUParameter rdf:datatype="#string">
          send(alert.InLanACE)=yes,severity.alert=low
        </hasUParameter>
        <hasUValue rdf:datatype="#float">-80</hasValue>
      </UtilityCell>
    </hasUCell>
    ...
    <hasUCell>
      <UtilityCell rdf:ID="cell_6">
        <hasUParameter rdf:datatype="#string">
          send(alert.InLanACE)=no,severity.alert=high
        </hasUParameter>
        <hasUValue rdf:datatype="#float">-100</hasValue>
      </UtilityCell>
    </hasUCell>
  </UtilityTable>
</owl:Class>
```

Fig. 13: Utility encoding

As seen in Figure 7, a sequential path between all decisions exists. Indeed, some decision depends on previous decisions and as a consequence, previous decisions (Decision node) become Chance nodes for next Chance node. Figure 12 illustrates that send(alert.InLanACE) is at the same time a Decision node and a Chance node that is known be the decision node alertForward2(InLanACE, UpperACE).

## IV. Conclusion

In this paper we have presented a global solution developed for an incident reaction system based on a policy regulation approach strategy. The solution is composed firstly with a MAS that offers the advantage to react quickly and efficiently against an attack while being adapted for distributed networks and secondly with a decision support system that helps agents to make decisions based on utility preference values. This is achieved by taking uncertainty into account through Bayesian networks and influence diagram.

The decision support system has been explained for the transfer of an alert from the alert correlation engine to the policy instantiation engine. Other decision points exist in the architecture. All of them could be solved using decision support system but they are not explained in the paper.

An important advantage of this decision support system is its capability to take decision at different points of the network. If more knowledge is needed to take a decision, the higher layer gives a global view of the whole system and is able to take decisions based on a more precise state of the system.

The future works based on our achievements will be the specification of a protocol, specification of the messages and thus the reaction methodology service oriented based. This protocol and methodology will be dedicated to the architecture presented in this paper and address the interoperability issues with regard to the policy representation and modeling.

## Acknowledgment

This research was funded by the National Research Fund of Luxembourg in the context of TITAN (Trust-Assurance for Critical Infrastructures in Multi-Agents Environments, FNR CO/08/IS/21) project.

## References

[1] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Commun.*, 7(1):39–59, 1994.

[2] Cataldo Basile, Antonio Lioy, Gregorio Martínez Pérez, Félix J. García Clemente, and Antonio F. Gómez-Skarmeta. Positif: A policy-based security management system. In *POLICY*, page 280. IEEE Computer Society, 2007.

[3] F. Bellifemine, G. Caire, A. Pogg, and G. Rimassa. Jade - a white paper. Technical Report 3, Telecom Italia Lab, EXP Online, 2003.

[4] Fabio Bellifemine, Agostino Poggi, and Giovanni Rimassa. Jade - a fipa-compliant agent framework. Technical report, CSELT, 1999.

[5] Elisa Bertino, Alessandra Mileo, and Alessandro Provetti. Pdl with preferences. In *POLICY*, pages 213–222. IEEE Computer Society, 2005.

[6] E. Bulut, D. Khadraoui, and B. Marquet. Multi-agent based security assurance monitoring system for telecommunication infrastructures. Network, and Information Security conference (CNIS 2007), Berkeley, California, USA, 2007.

[7] John Butler, Douglas J. Morrice, and Peter W. Mullarkey. A multiple attribute utility theory approach to ranking and selection. *Manage. Sci.*, 47(6):800–816, 2001.

[8] Carlos Carrascosa, Javier Bajo, Vicente Julián, Juan M. Corchado, and Vicente J. Botti. Hybrid multi-agent architecture as a real-time problem-solving model. *Expert Syst. Appl.*, 34(1):2–17, 2008.

[9] A. Cuevas, P. Serrano, J. I. Moreno, C. J. Bernardos, J. Jähnert, and R. L. Aguiar. Usability and evaluation of a deployed 4g network prototype. *Journal of Communications and Networks*, Vol. 7 (2), 2008.

[10] Nicodemos Damianou, Naranker Dulay, Emil Lupu, and Morris Sloman. The ponder policy specification language. In Morris Sloman, Jorge Lobo, and Emil Lupu, editors, *POLICY*, volume 1995 of *Lecture Notes in Computer Science*, pages 18–38. Springer, 2001.

[11] H. Debar, France Telecom, D. Curry, Guardian, B. Feinstein, and Inc. SecureWorks. The intrusion detection message exchange format (idmef). *IDMEF/RFC4765*, 2007.

[12] C. Feltus, D. Khadraoui, B. de Rémont, and A.Rifaut. Business gouvernance based policy regulation for security incident response. *IEEE Global Infrastructure Symposium*, 2007.

[13] FIPA. http://www.fipa.org/.
[14] B. Gâteau. *Modélisation et Supervision d'Institutions Multi-Agents*. PhD thesis, École Supérieure des Mines de Saint-Étienne, 2007.
[15] B. Gâteau, D. Khadraoui, and C. Feltus. Multi-agents system service based platform in telecommunication security incident reaction. *IEEE Global Information Infrastructure Symposium*, 2009.
[16] Benjamin Gâteau, Djamel Khadraoui, Christophe Feltus, and Benoît De Rémont. Multi-agents based architecture for is security incident reaction. In *RIVF*. [PDF], 2008.
[17] R. A. Howard and J.E. Matheson. Influence diagrams. *Decision Analysis*, pages 127–143, 2005.
[18] Finn V. Jensen. *Bayesian Networks and Decision Graphs (Information Science and Statistics)*. Springer, July 2001.
[19] L. J. LaPadula. State of the art in anomaly detection and reaction technical report mp 99b0000020. *Mitre*, July 1999.
[20] H. D. Lasswell. The decision process; seven categories of functional analysis. *College of Business and Public Administration, University of Maryland*, 1956.
[21] Kun-Yung Lu and Chun-Chin Sy. A real-time decision-making of maintenance using fuzzy agent. *Expert Syst. Appl.*, 36(2):2691–2698, 2009.
[22] M. Petkac and Lee Badger. Security agility in response to intrusion detection. In *ACSAC*, pages 11–20. IEEE Computer Society, 2000.
[23] G.L.F. Santos, Z. Abdelouahab, R.A. Dias, C.F.L. Lima, E. Nascimento, and E.M. Cochra. An automated response approach for intrusion detection security enhancement. *Software Engineering and Applications*, 2003.
[24] Rudi Studer, V. Richard Benjamins, and Dieter Fensel. Knowledge engineering: Principles and methods. *Data Knowl. Eng.*, 25(1-2):161–197, 1998.
[25] J.A. Tatman and R.D. Shachter. Dynamic programming and influence diagrams. *IEEE Transactions on Systems, Man, and Cybernetics*, 20(2):365–379, 1990.
[26] Joseph Chee Ming Teo, Chik How Tan, and Jim Mee Ng. Denial-of-service attack resilience dynamic group key agreement for heterogeneous networks. *Telecommunication Systems*, 35(3-4):141–160, 2007.
[27] Bayes theorem. http://plato.stanford.edu/entries/bayes-theorem/.
[28] Gustavo A. Santana Torrellas and Luis A. Villa Vargas. Modelling a flexible network security systems using multi-agents systems: security assessment considerations. In *ISICT*, volume 49 of *ACM International Conference Proceeding Series*, pages 365–371. Trinity College Dublin, 2003.
[29] Y. Yang. *A framework for decision support systems adapted to uncertain knowledge*. PhD thesis, University of Karlsruhe, 2007.
[30] R. Yu, B. Iung, and H. Panetto. A multi-agents based e-maintenance system with case-based reasoning decision support. Engineering Applications of Artificial Intelligence, pages 321–333.